# HHN의 개선된 routing algorithm

*김우영°, *이미선, **이형옥, *허영남
*순천대학교 컴퓨터과학과
**순천대학교 컴퓨터교육과
e-mail:*chamoim°@netian.com
{*lms, **oklee, *hyn}@sunchon.ac.kr

# Improved routing algorithm of HHN(Hierarchical Hypercube Networks)

Woo-Young Kim° Mi-Sun Lee Hyeong-Ok Lee Yeong-Nam Heo
Dept. of Computer Science, Sunchon National University
Dept. of Computer Education, Sunchon National University

요약

최근에 대규모 병렬컴퓨터의 새로운 위상으로 *HHN(m;n,h)* 그래프가 제안되었다. *HHN(m;n,h)* 그래프는 하이퍼큐브에 기반한 계층적인 상호 연결망이며, 노드개수가 유사한 하이퍼큐브보다 작은 노드 분지수와 작은 링크의 수를 가지고 있기 때문에 대규모 병렬 컴퓨터를 구성하는 상호 연결망이지만, 심플 라우팅 알고리즘에 기반한 경로를 구성함으로써 불필요한 라우팅 경로를 갖는 단점이 있다. 따라서 본 논문에서는 심플 라우팅 알고리즘보다 평균거리가 개선된 새로운 라우팅 알고리즘을 제안하고, *HHN*(3;3,2)를 기준으로 비교하였을 때 대략 32%정도 향상됨을 보인다.

## 1. 서론

고성능 병렬 컴퓨터 개발의 필요성이 날로 증대함에 따라 이러한 병렬 컴퓨터의 성능에 중요한 영향을 미치는 상호 연결망에 대한 연구 또한 활발하게 진행되어 왔다.

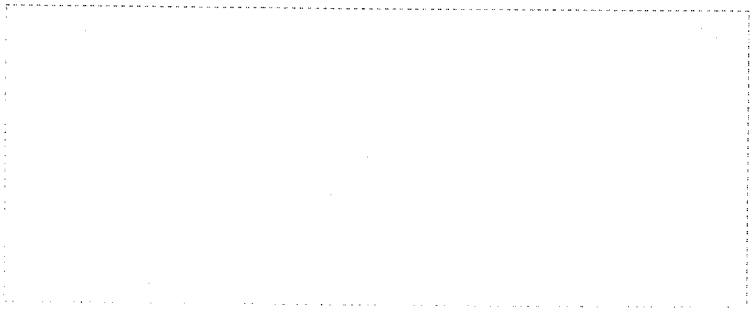
상호 연결망은 각 프로세서를 노드로, 프로세서들 사이에 통신 채널을 에지로 나타내는 무방향 그래프로 표현되는데, 지금까지 제안된 상호연결망은 노드 개수를 기준으로 분류하면 $k \times n$으로 표현되는 메쉬 부류, $2^n$으로 표현되는 하이퍼큐브(hypercube) 부류, $n!$로 표현되는 스타(star) 그래프 부류로 나눌 수 있다. 상호 연결망을 평가하는 척도로는 분지수(degree), 연결도(connectivity), 대칭성(symmetric), 지름(diameter), 평균거리(average distance), 고장지름(fault diameter), 망비용(network cost), 방송(broadcasting), 임베딩(embedding) 등이 있다[3, 4].

하이퍼큐브 연결망은 노드 및 에지 대칭이고 심플한 재귀적 구조를 가지고 있어서 각종 응용분야에서 요구하는 통신망 구조를 쉽게 제공할 수 있는 장점이 있어 기존의 연구용 및 상용 시스템에 널리 사용되고 있다. 또한, 임베딩 관점에 있어서 링, 트리, 피라미드, 메쉬 등과 같은 다른 연결망 구조들이 효율적으로 임베딩 될 수 있다는 장점이 있지만, 분지수에 비해 지름과 노드간의 평균거리가 짧지 않다는 단점이 있다. 이러한 문제를 극복하기 위해 노드와 에지의 인접관계를 조작하여 하이퍼큐브의 연결망을 망비용 측면에서 개선한 새로운 연결망으로 하이퍼넷(Hypernet)[1], HCN(Hierarchical Cubic Network)[2], HHN(Hierarchical Hypercube Networks)[5] 등이 제안되었다.

HHN은 대규모 병렬 컴퓨터의 새로운 위상으로 제안된 그래프로써 하이퍼큐브에 기반한 계층적인 상호 연결망이며, 노드개수가 유사한 하이퍼큐브 보다 작은 노드 분지수와 작은 링크의 수를 가지고 있기 때문에 대규모 병렬 컴퓨터를 구성하는 아주 뛰어난 상호 연결망으로 대규모 병렬 시스템을 위하여 널리 보급되어 있는 하이퍼큐브 보다 우수한 연결망이다. 라우팅 알고리즘을 설정할 때 심플 라우팅 알고리즘을 사용한다.

그러나, 심플 라우팅 알고리즘에 기반한 경로를 구성함으로써 불필요한 라우팅 경로를 많이 갖고 있어서 비효율적이다.

따라서 본 논문에서는 심플 라우팅 알고리즘보다 평균거리가 개선된 라우팅 알고리즘을 제안하고, 최



(그림 1) HHN(3;3,1) and HHN(3;3,2) network

악의 경로 값은 같지만 평균거리 값이 <표3>처럼 향상되었다.

제2장은 관련연구로 HHN의 네트워크의 구조 및 특징을, 제3장은 HHN의 심플 라우팅 알고리즘과 문제점을, 제4장은 HHN의 개선된 라우팅 알고리즘을, 마지막으로 결론을 맺도록 하겠다.

## 2. 관련연구

HHN은 클러스터라 불리는 하이퍼큐브 내에서 노드들 간에 연결하는 에지를 내부 에지라 하고, 클러스터와 클러스터간을 연결하는 에지를 외부 에지라 한다. 그림.1.은 HHN의 구조를 가장 하위레벨인 하이퍼큐브에서부터 HHN의 2차원 레벨의 모습을 계층적으로 잘 보여주고 있다.

$HHN(m;n,h)$의 정의는 다음과 같다($m \le n$) :

$m$ : $2^m$ 개의 클러스터 개수

$n$ : $n$-차원의 하이퍼큐브

$h$ : 차원(=레벨)

각 노드는 $(m+nh)$개의 이진수로 구성되고, 노드 주소는 $(A_h, \cdots, A_1, A_0)$로 표시한다. 노드 개수는 $2^{m+nh}$이고, 분지수는 $n+h$이며, 에지수는 $(n+h)2^{m+nh-1} - h2^{m+(h-1)n-1}$ 이고, 지름은 $m+hn+h$ 이고, 평균 거리는 $(m+nh)/2+h-(m/2+1)/2^m-(h-1)(n/2+1)/2^n$ 이다.

내부 에지는 하이퍼큐브의 정의에 의해 발생하고, 외부 에지는 다음과 같이 2가지 조건에 의해 발생한다.

1. $1 \le j < h$일 때, $A_j \neq A_0$이면, 노드 $(\cdots, A_{j+1}, A_j, A_{j-1}, \cdots, A_1, A_0)$은 노드 $(\cdots, A_{j+1}, A_0, A_{j-1}, \cdots, A_1, A_j)$에 $j$번째 외부 에지 $L_j$를 통하여 연결된다.

2. $A_h \neq A_0^L$이면, 노드$(A_h, A_{h-1}, \cdots, A_1, A_0)$은 노드$(A_0^L, A_{h-1}, \cdots, A_1, A_0^H|A_h)$에 $h$번째 외부에지 $L_h$를 통하여 연결된다.

$A_0^H$은 상위의 $(n-m)$비트를 표현하고,

$A_0^L$은 $A_0$의 하위의 $m$-비트를 표현한다.

$A_0^H|A_h$는 2개의 수 $A_0^H$와 $A_h$의 연결이다.

## 3. HHN의 심플 라우팅 알고리즘과 문제점

$HHN(m;n,h)$안의 임의의 두 노드를 $u=(A_h, ..., A_1, A_0)$과 $v=(B_h, ..., B_1, B_0)$라 하면, 노드 $u$에서 노드 $v$로의 라우팅은 다음과 같다.

$j<h$이면, 경로는

$(\cdots, A_{j+1}, A_j, A_{j-1}, \cdots, A_1, A_0) \Rightarrow (\cdots, A_{j+1}, A_j, A_{j-1}, \cdots, A_1, B_j) \rightarrow (\cdots, A_{j+1}, B_j, A_{j-1}, \cdots, A_1, A_j)$이고,

$j=h$이면, 경로는

$(A_h, A_{h-1}, \cdots, A_1, A_0) \Rightarrow (A_h, A_{h-1}, \cdots, A_1, A_0^H|B_h) \rightarrow (B_h, A_{h-1}, \cdots, A_1, A_0^H|A_h)$이다.

여기서, $\Rightarrow$은 클러스터안에서의 내부 라우팅 경로를 표현하고, $\rightarrow$은 클러스터와 클러스터간의 외부 라우팅 경로를 표현한다.

예를 들면, $HHN(4;4,3)$안에 $u$=(0010, 0100, 0011, 1001)에서 $v$=(0110, 0011, 1000, 0111)까지 라우팅 경로를 고려할 때 심플 라우팅 알고리즘에 의한 라우팅 경로는 다음과 같다($\Rightarrow$옆의 숫자는 내부 라우팅 할 때의 비트 수를 의미한다) :

(0010, 0100, 0011, 1001) $\Rightarrow$4 (0010, 0100, 0011, 0110) $\rightarrow$ (0110, 0100, 0011, 0010) $\Rightarrow$1 (0110, 0100, 0011, 0011) $\rightarrow$ (0110, 0011, 0011, 0100) $\Rightarrow$2 (0110, 0011, 0011, 1000) $\rightarrow$ (0110, 0011, 1000, 0011) $\Rightarrow$1 (0110, 0011, 1000, 0111).

따라서 경로 값을 살펴보면, 내부 라우팅 8회와 외부 라우팅 3회로 전체 라우팅 횟수는 11회이다. 즉, $u$ 노드에서 $v$ 노드까지 11회의 라우팅 경로 값을 가진다.

위의 예에서 살펴보듯이 목적노드 $v$의 최상위 레벨($B_h$)부터 최하위 레벨($B_0$)까지 감소하는 레벨의 순서에 따라 정정되어지는 비최적화, 그러나 심플 라우팅 알고리즘이 제안되었었다. 이러한 심플 라우팅 알고리즘으로 말미암아 불필요한 라우팅 경로 값을 갖게 되어 비교되는 하이퍼큐브 보다 작은 링크의 수를 가지고 있는 만큼의 최적의 라우팅 경로 값을 갖지 못한다. 따라서 다음 장에서 개선된 라우팅 알고리즘을 제안한다.

## 4. HHN의 개선된 라우팅 알고리즘

임의의 두 노드를 $u=(A_h, \cdots, A_1, A_0)$, $v=(B_h, \cdots, B_1, B_0)$라 하고, ⇒은 하이퍼큐브 내에서의 라우팅을 나타내고, →은 외부 링크 라우팅을 나타내며, 튜플은 노드 $u$, $v$를 구성하는 각각의 원소를 나타낸다.

개선된 라우팅 알고리즘은 다음과 같은 3개의 step에 의해 경로를 설정하게 된다.

**step1** : 두 노드 $u$와 $v$를 행과 열의 원소가 튜플 단위로 구성된 matrix에서 각각 행과 열에 상응하는 값들을 XOR(⊕)한 후 1의 개수를 matrix의 구성원소로 사용한다.

**step2 : pair 선택**

· pair : 선택된 구성원소의 행과 열의 튜플 값

· $p$ : matrix 내의 구성원소로 pair된 튜플 값의 ⊕된 1의 개수.

· $D$(diagonal) : matrix 내 구성원소들의 대각선으로 다음과 같다.

$$D=( a_hb_h, a_{h-1}b_{h-1}, \cdots, a_1b_1, a_0b_0 )$$

· $A_k$ : 동일한 값을 갖는 구성원소들 중 최상위 레벨의 튜플 값.

1. $D$의 구성원소 중 $p$=0인 것을 pair로 선택하고, 선택된 구성원소가 속한 열과 행을 제거한다.
2. $D$를 제외한 모든 영역에서 $p$=0인 것을 찾아 pair로 선택하고, 선택된 구성원소가 속한 열과 행을 제거한다. 단, $p$=0인 동일한 값이 존재하면 $A_k$의 값을 선택한다.
3. 대각선 $D$에 위치한 값에 각각 1씩 더한다.
4. $B_0$열중 나머지 값 가운데 가장 작은 구성원소를 선택하고, 그 열과 행을 제거한다. 단 동일한 값이 존재하면 $A_k$의 값을 선택한다.
5. pair 지어진 값 외의 모든 영역에서 가장 작은 값들을 선택하여 차례로 pair를 지어주고, 그 열과 행을 제거하되, $B_0$열에서 $B_h$열 순으로 한다. 단, 주어진 열의 값 가운데 동일한 값이 존재하면 $A_k$값을 선택한다.
6. pair로 모두 선택되면, 대각선 $D$에 1을 더한 값을 다시 뺀 후 step3과 같이 라우팅을 시작한다.

**step3 : 라우팅**

1. 시작노드 u의 $A_0$부터 pair 지어진 순서대로 아래와 같이 라우팅 한다.

**if** $A_0$와 pair된 값이 대각선 $D$의 값으로 $p$=0이면, 값이 같은 자기 자리이므로 라우팅하지 않는다.

**elseif** 대각선 $D$가 아니면서 $A_0$와 pair된 값이 $p$=0이면, →(외부 라우팅)으로 $A_0$와 pair된 $B_x$열의 주어진 값을 $u$노드와의 동일한 레벨 값과 교환한다.

**else**

$p$만큼 $A_0$와 pair된 $B_x$열의 주어진 값으로 ⇒(내부 라우팅) 하고, 그 값을 동일한 레벨 값으로 →(외부 라우팅) 하여 교환한다.

그러나, 그 값이 $v$노드의 최하위 레벨 값($B_0$)이면, (라우팅 하지 않은 $u$노드 중) $A_0$의 상위 레벨 값과 →(외부 라우팅)으로 교환한다.

2. 목적노드 $v$까지 계속 반복한다.

앞의 3장에 든 예로써 예를 들면, $HHN(4;4,3)$안에 $u$=(0010, 0100, 0011, 1001)에서 $v$=(0110, 0011, 1000, 0111)까지 라우팅 경로를 고려할 때 본 논문에서 제안한 개선된 라우팅 알고리즘에 의한 step1부터 step2까지의 과정을 표1을 통하여 셀 마킹 표시로 노드간에 pair된 상태를 보여준다. 또한, 점(.)이 표시된 부분은 심플 라우팅 알고리즘에 의해 노드간에 서로 선택된 상태를 비교적으로 보여준다.

<표 1> 개선된 알고리즘의 step1과 step2의 과정

| ⊕ | | $b_h$ | $b_2$ | $b_1$ | $b_0$ |
|---|---|---|---|---|---|
| | | 0110 | 0011 | 1000 | 0111 |
| $a_h$ | 0010 | $1_2$ | 1. | 2 | 2 |
| $a_2$ | 0100 | 1 | $3_4$ | 2. | 2 |
| $a_1$ | 0011 | 2 | 0 | $3_4$ | 1. |
| $a_0$ | 1001 | 4. | 2 | 1 | $3_4$ |

step3에 의한 라우팅 경로의 과정은 아래와 같다 (⇒옆의 숫자는 내부 라우팅 할 때의 비트 수를 의미한다).

(0010, 0100, 0011, 1001) ⇒1 (0010, 0100, 0011, 1000) → (0010, 0100, 1000, 0011) → (0010, 0011, 1000, 0100) ⇒1 (0010, 0011, 1000, 0110) → (0110, 0011, 1000, 0010) ⇒2 (0110, 0011, 1000, 0111)

따라서 경로 값을 살펴보면, 내부 라우팅 4회와 외부 라우팅 3회로 전체 라우팅 횟수는 7회이다. 즉, $u$ 노드에서 $v$ 노드까지 7회의 라우팅 경로 값으로 횟수로는 4회를 줄였고, 약36%의 라우팅 경로 값을 개선하게 되었다.

본 논문에서 예로 든 두 노드 $u$=(0010, 0100, 0011, 1001)와 $v$=(0110, 0011, 1000, 0111)에 대한 두 알고리즘의 라우팅 결과 값을 비교하면 표2와 같다.

<표 2> 주어진 예에서 두 알고리즘의 경로 값 비교표

| 구 분 | | 심플 라우팅 알고리즘의 경로 값 | 개선된 알고리즘의 경로 값 |
|---|---|---|---|
| 예 | 내부 라우팅값 | 8 | 4 |
| | 외부 라우팅값 | 3 | 3 |
| | 총 라우팅 경로값 | 11 | **7(36%)** |

최악의 경우, 예를 들면 노드 $u$=(1111, 1111, 1111, 1111)에서 노드 $v$=(0000, 0000, 0000, 0000)로 라우팅 할 경우의 경로 값은 두 라우팅 알고리즘에서 모두 내부 라우팅 16회와 외부 라우팅 3회로 총 19회의 라우팅 횟수를 보인다.

또한, *HHN*(2;2,2), *HHN*(3;3,2), *HHN*(4;4,3), *HHN*(5;5,4)에서 임의의 두 노드에 대한 개선된 라우팅 알고리즘의 평균 경로 값과 심플 라우팅 알고리즘의 경로 값을 비교한 결과는 표3과 같다.

단, *HHN*(2;2,2)와 *HHN*(3;3,2)는 모든 경우의 수를 다 참조하였고, *HHN*(4;4,3)는 경우의 수가 너무 많아 노드 $u$의 하나 값을 고정시킨 후 노드 $v$의 모든 값을 다 참조하였으며, *HHN*(5;5,4)는 노드 $u$의 하나 값에 노드 $v$의 경우의 수에 1/100값(335,544가지)만 추출하여 참조한 결과 얻은 값이다.

<표 3> ***HHN*(2;2,2), *HHN*(3;3,2), *HHN*(4;4,3), *HHN*(5;5,4)에서 라우팅 알고리즘의 경로 값 비교**

| 구 분 | 심플 라우팅 알고리즘 | **개선된 라우팅 알고리즘** | |
|---|---|---|---|
| | 평균거리 | **평균거리** | **개선 값** |
| *HHN*(2;2,2) | 4 | **2.5** | **37.5%** |
| *HHN*(3;3,2) | 5.875 | **4** | **32%** |
| *HHN*(4;4,3) | 10.438 | **8** | **23%** |
| *HHN*(5;5,4) | 16.063 | **12** | **25%** |

따라서, 표3에서 보인 결과처럼 심플 라우팅 알고리즘과 개선된 라우팅 알고리즘의 라우팅 경로 값을 비교한 결과 심플 라우팅 알고리즘 보다 본 논문의 개선된 라우팅 알고리즘이 *HHN*(3;3,2)의 경우 평균거리의 값이 32% 정도 향상됨을 보인다.

## 5. 결론

*HHN*(*m*;*n*,*h*) 그래프는 하이퍼큐브에 기반한 계층적인 상호 연결망이며, 노드개수가 유사한 하이퍼큐브 보다 작은 노드 분지수와 작은 링크의 수를 가지고 있기 때문에 대규모 병렬 컴퓨터를 구성하는 상호 연결망이지만, 심플 라우팅 알고리즘에 기반한 경로를 구성함으로써 불필요한 라우팅 경로를 갖는 단점이 있다. 따라서 본 논문에서는 심플 라우팅 알고리즘보다 평균거리가 개선된 새로운 라우팅 알고리즘을 제안한다.

본 논문에서 제안한 개선된 라우팅 알고리즘과 심플 라우팅 알고리즘을 비교하였을 때, 최악의 경우 평균거리의 값은 같지만, *HHN*(3;3,2)의 경우에서는 심플 라우팅 알고리즘의 평균거리 값이 5.875인 것을 개선된 라우팅 알고리즘의 평균거리 값이 4로 32%정도 향상된 결과를 가져왔다.

향후 연구 과제는 최적 라우팅 알고리즘을 찾는 것이다.

## 참고문헌

[1] K. Ghose and K. R. Desai "Hierarchical Cubic Networks," IEEE Trans. Parallel Distributed syst., Vol.6, No. 4, pp.427-436, 1995.

[2] K. Hwang, and J. Ghosh, "Hypernet:A communication efficient architecture for constructing massively parallel computers," IEEE Trans. Comput. C-36, 12(Dec. 1987), 1450-1466.

[3] F. T. Leighton, Introduction to Parallel Algorithms and Architectures : Arrays, Hypercubes, Morgan Kaufmann Publishers, 1992.

[4] V. E. Mendia nad D. Sarkar, "Optimal Broadcasting on the Star Graph," IEEE Trans. Parallel Distributed syst., Vol.3, No.4, pp. 389-396, 1992.

[5] Sang-Kyun Yun and Kyu Ho Park, Hierarchical Hypercube Networks(HHN) for Massively parallel Computers. Journal of Parallel and Distributed Computing 37, 194_199(1996). article no. 0119.